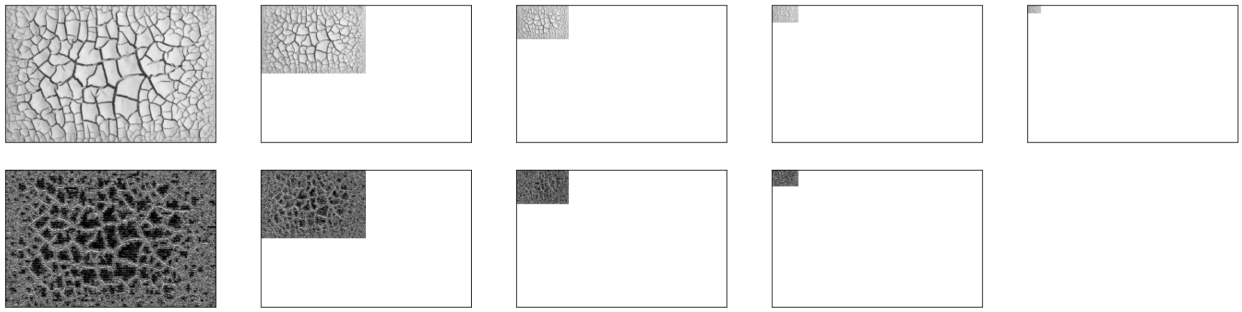
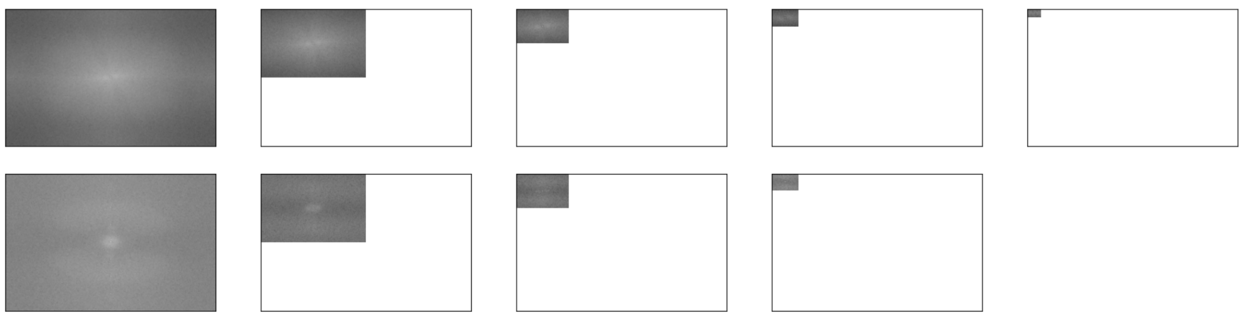
Gaussian/Laplacian Pyramid



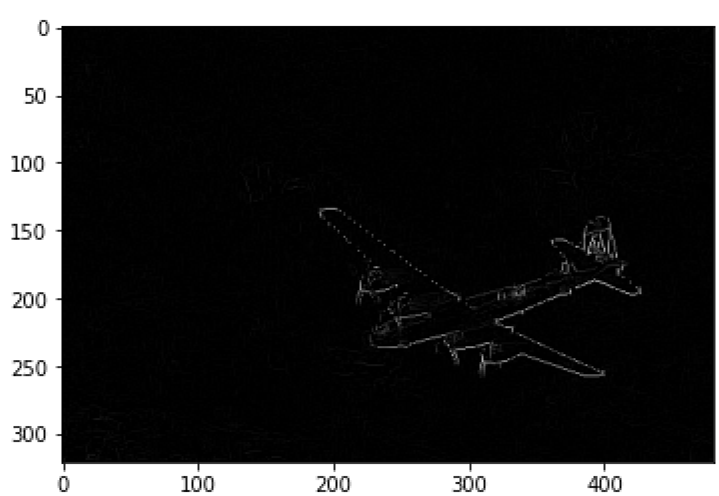
Fourier Transform of Pyramid



Note: I used np.log() of the np.abs() of the resultant values instead of finding vmin and vmax values to properly display the FFT images.

P1.2 Explanation:
The Gaussian filter reduces noise / high frequency components in the image. Its Fourier shifted version has lower frequency components/less high freq. components than the original's F-shifted plot would.

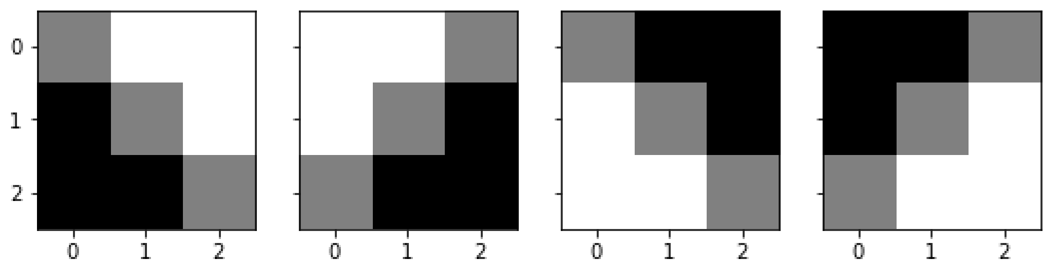
The Laplacian is just the residual difference from the original image at that level and that level's gauss-blurred counterpart. The frequency is spread more throughout the image, as can be seen visually on the L-pyramid's Fourier shifted plot, and this is because high-frequency components were reduced throughout the entire image by the Gaussian filter -- the frequencies that were reduced all over are being revealed in the Laplacian pyramid.



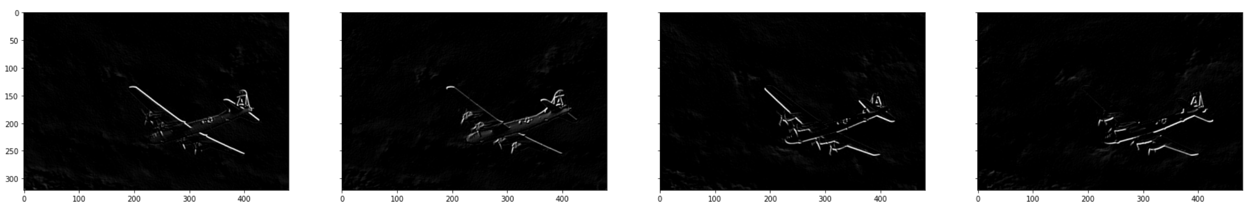
This is the **Problem 2.1** Resulting boundary map

Problem 2.1 Note:
I ran into the problem of overflow when manually computing the L2 norm of gradients and had to make sure everything was converted to int64 type – it was interesting that this came up.

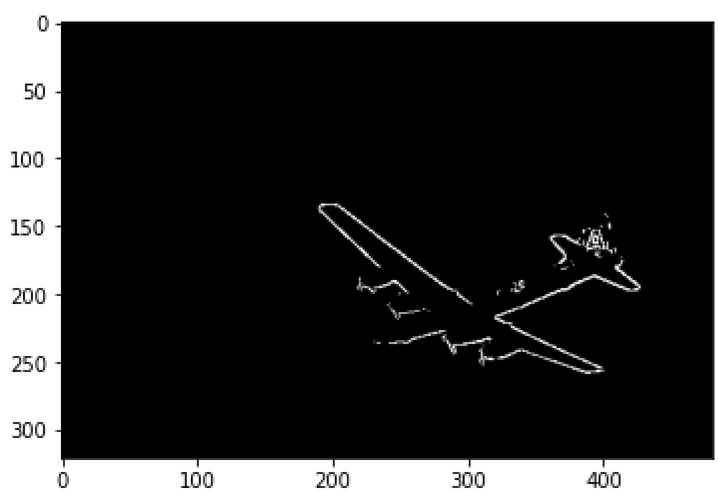
Problem 2.2 Oriented FIlters Visualized

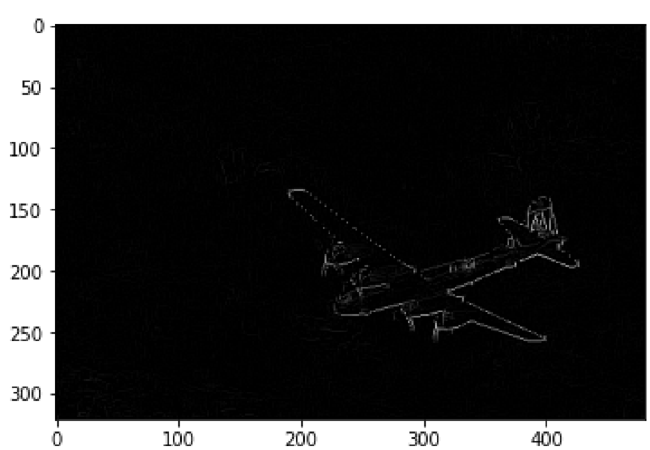


Problem 2.2 Oriented Filters individual effect on image



Problem 2.2 Resulting Post- NM Suppression image



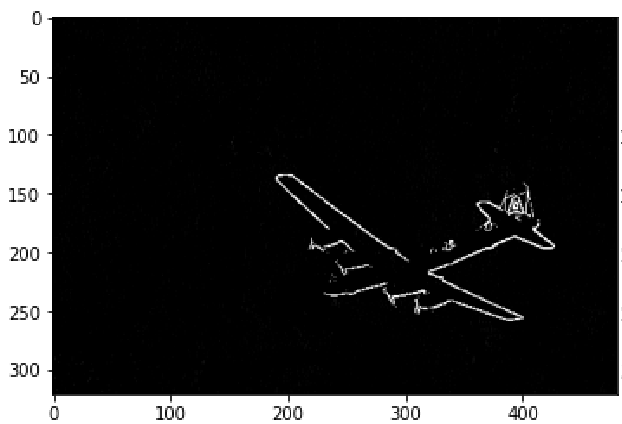


Compare back to the 2.1 Result:
Clearly a better boundary map was achieved on the above image with oriented filters.

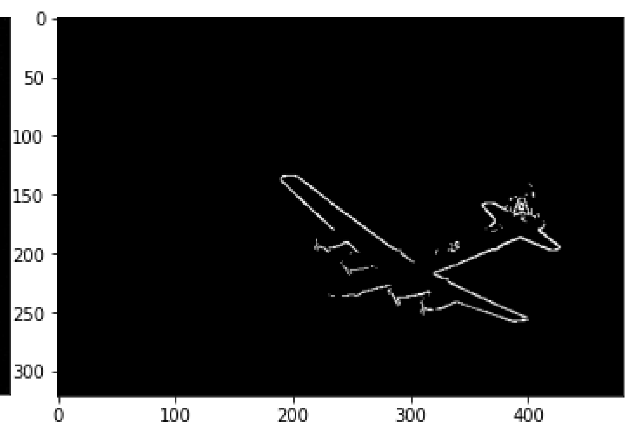
Problem 2.3 Idea:

One method of improvement is to re-run with more orientations, e.g. one orientated filter for every 15 degrees instead of the 4 diagonals that I used at first. I re-ran the above functions with the angle set to be one for every 15 degrees, and found that boundary map to include slightly more defined edges. That image is attached and shown with the 4-way oriented filters image result for comparison. The most noticeable area of improvement is the ‘A’ paint area on the tail of the plane.

15-Filter Result



4-Filter Result



The partner of this for the non-max suppression would be to separate the 'bins' for each angle to be more granular i.e. many smaller ranges, and use linear interpolation of adjacent gradients to come up with the gradient magnitude at each spot that is neither orthogonal nor perfectly diagonal. This would allow for more precise non-max-suppression. This should help because when the angle is rounded to aim at a directly adjacent gradient magnitude, that magnitude's pixel is more likely to actually be part of a different edge that is being 'rounded into' this edge, leading to suppressions that could be incorrect.

Combining both of these techniques would be a good way to improve both the boundary scoring, and the precision of the suppression technique.